1. Giới thiệu

Ngôn ngữ mô hình hóa thống nhất (tiếng Anh: Unified Modeling Language, viết tắt thành UML) là một ngôn ngữ mô hình gồm các ký hiệu đồ họa mà các phương pháp hướng đối tượng sử dụng để thiết kế các hệ thống thông tin một cách nhanh chóng.

Cách xây dựng các mô hình trong UML phù hợp mô tả các hệ thống thông tin cả về cấu trúc cũng như hoạt động. Cách tiếp cận theo mô hình của UML giúp ích rất nhiều cho những người thiết kế và thực hiện hệ thống thông tin cũng như những người sử dụng nó; tạo nên một cái nhìn bao quát và đầy đủ về hệ thống thông tin dự định xây dựng. Cách nhìn bao quát này giúp nắm bắt trọn vẹn các yêu cầu của người dùng; phục vụ từ giai đoạn phân tích đến việc thiết kế, thẩm định và kiểm tra sản phẩm ứng dụng công nghệ thông tin. Các mô hình hướng đối tượng được lập cũng là cơ sở cho việc ứng dụng các chương trình tự động sinh mã trong các ngôn ngữ lập trình hướng đối tượng, chẳng hạn như ngôn ngữ C++, Java,... Phương pháp mô hình này rất hữu dụng trong lập trình hướng đối tượng. Các mô hình được sử dụng bao gồm Mô hình đối tượng (mô hình tĩnh) và Mô hình động.

UML sử dụng một hệ thống ký hiệu thống nhất biểu diễn các Phần tử mô hình (model elements). Tập hợp các phần tử mô hình tạo thành các sơ đồ UML (UML diagrams). Có các loại sơ đồ UML chủ yếu sau:

* Sơ đồ lớp (Class Diagram)
* Sơ đồ đối tượng (Object Diagram)
* Sơ đồ tình huống sử dụng (Use Cases Diagram)
* Sơ đồ trình tự (Sequence Diagram)
* Sơ đồ cộng tác (Collaboration Diagram hay là Composite Structure Diagram)
* Sơ đồ trạng thái (State Machine Diagram)
* Sơ đồ thành phần (Component Diagram)
* Sơ đồ hoạt động (Activity Diagram)
* Sơ đồ triển khai (Deployment Diagram)
* Sơ đồ gói (Package Diagram)
* Sơ đồ liên lạc (Communication Diagram)
* Sơ đồ tương tác (Interaction Overview Diagram - UML 2.0)
* Sơ đồ phối hợp thời gian (Timing Diagram - UML 2.0)

1. Một số dạng sơ đồ UML phổ biến
   1. Sơ đồ Use case (Use case diagram)

Một sơ đồ Use case chỉ ra một số lượng các tác nhân ngoại cảnh và mối liên kết của chúng đối với Use case mà hệ thống cung cấp. Một Use case là một lời miêu tả của một chức năng mà hệ thống cung cấp. Lời miêu tả Use case thường là một văn bản tài liệu, nhưng kèm theo đó cũng có thể là một Sơ đồ hoạt động. Các Use case được miêu tả duy nhất theo hướng nhìn từ ngoài vào của các tác nhân (hành vi của hệ thống theo như sự mong đợi của người sử dụng), không miêu tả chức năng được cung cấp sẽ hoạt động nội bộ bên trong hệ thống ra sao. Các Use case định nghĩa các yêu cầu về mặt chức năng đối với hệ thống.

* Hệ thống: Với vai trò là thành phần của sơ đồ use case, hệ thống biểu diễn ranh giới giữa bên trong và bên ngoài của một chủ thể trong phần mềm chúng ta xây dựng.
* Tác nhân(actor):là người dùng của hệ thống, một tác nhân có thể là một người dùng thực hoặc các hệ thống máy tính khác có vai trò nào đó trong hoạt động của hệ thống. Như vậy, tác nhân thực hiện các use case. Một tác nhân có thể thực hiện nhiều use case và ngược lại một use case cũng có thể được thực hiện bởi nhiều tác nhân.

Kí hiệu



Figure 1: Tác nhân



Figure 2 Use case

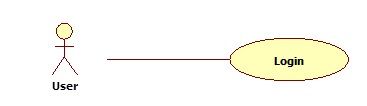


Figure 3 Association

Include: là quan hệ giữa các Use Case với nhau, nó mô tả việc một Use Case lớn được chia ra thành các Use Case nhỏ để dễ cài đặt (module hóa) hoặc thể hiện sự dùng lại.

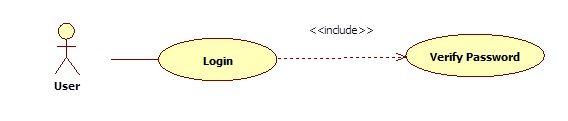


Figure 4 Quan hệ Include

Extend dùng để mô tả quan hệ giữa 2 Use Case. Quan hệ Extend được sử dụng khi có một Use Case được tạo ra để bổ sung chức năng cho một Use Case có sẵn và được sử dụng trong một điều kiện nhất định nào đó.

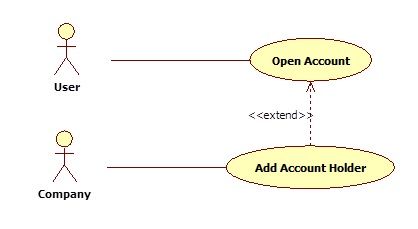


Figure 5 Quan hệ extent

Generalization: được sử dụng để thể hiện quan hệ thừa kế giữa các Actor hoặc giữa các Use Case với nhau.

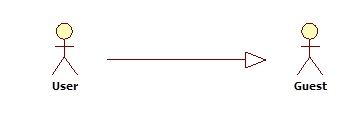


Figure 6 Generalization

* 1. Sơ đồ lớp (Class Diagram)

Một biểu đồ lớp chỉ ra cấu trúc tĩnh của các lớp trong hệ thống. Các lớp là đại diện cho các “đối tượng” được xử lý trong hệ thống. Các lớp có thể quan hệ với nhau trong nhiều dạng thức:

* liên kết (associated - được nối kết với nhau),
* phụ thuộc (dependent - một lớp này phụ thuộc vào lớp khác),
* chuyên biệt hóa (specialized - một lớp này là một kết quả chuyên biệt hóa của lớp khác),
* hay đóng gói ( packaged - hợp với nhau thành một đơn vị).
  + 1. Một lớp có các thành phần sau
  + Tên lớp
  + Các thuộc tính
  + Các phương thức

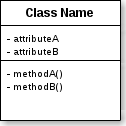


Figure 7 Thành phần của một lớp

* + 1. Liên kết giữa các lớp

Liên kết (Asociation):

* + Mối liên hệ ngữ nghĩa giữa hai hay nhiều lớp chỉ ra sự liên kết giữa các thể hiện của chúng.
  + Mối quan hệ về mặt cấu trúc chỉ ra các đối tượng của lớp này có kết nối với các đối tượng của lớp khác.

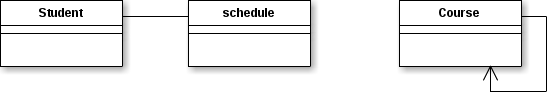


Figure 8 Association

Bội số quan hệ: là số lượng thể hiện của một lớp liên quan tới một thể hiện của lớp khác. Với mỗi liên kết, có hai bội số quan hệ cho hai đầu của liên kết.

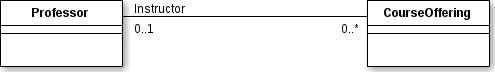


Figure 9 Bội số quan hệ

Biểu diễn bộ số quan hệ:

| **Biểu diễn** | **Ý nghĩa** |
| --- | --- |
|  | Unspecified |
| 1 | chính xác 1 |
| 0..\* | 0 hoặc nhiều |
| \* | 0 hoặc nhiều |
| 1..\* | 1 hoặc nhiều |
| 0..1 | 0 hoặc 1 |
| 2..4 | Specified Range |
| 2, 4..6 | Multiple, Disjoint Ranges |

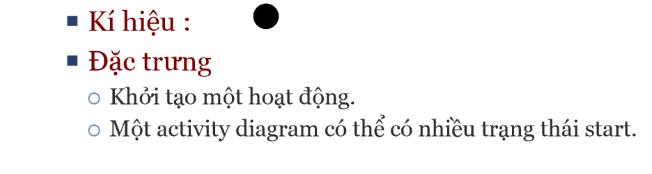
* 1. Sơ đồ hoạt động

**Activity Diagram** là một mô hình logic dùng để mô hình hoá các hoạt động trong một quy trình nghiệp vụ. Hay có thể hiểu **Activity – Diagram** là sơ đồ luồng xử lý của hệ thống. Bao gồm luồng đi của dòng dữ liệu, dòng sự kiện.

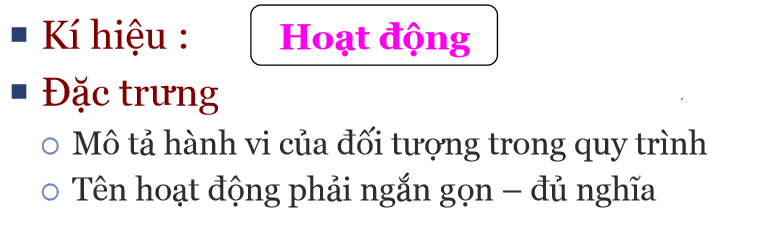
Dùng để mô tả các hoạt động trong một chức năng của hệ thống, có thể hiểu là mô tả luồng xử lý của một **Use – Case**.

Các thành phần của Activity – Diagram:

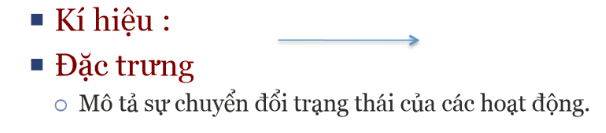
* + Start:.



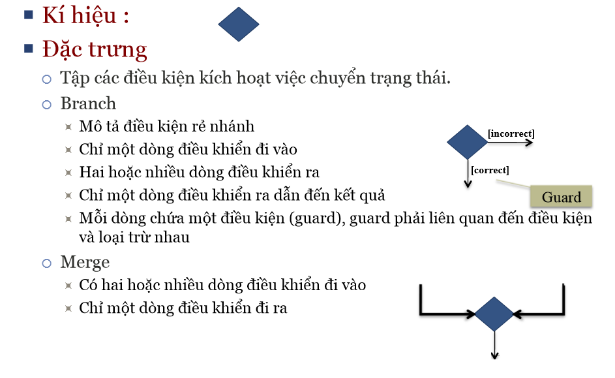
* + Activity:

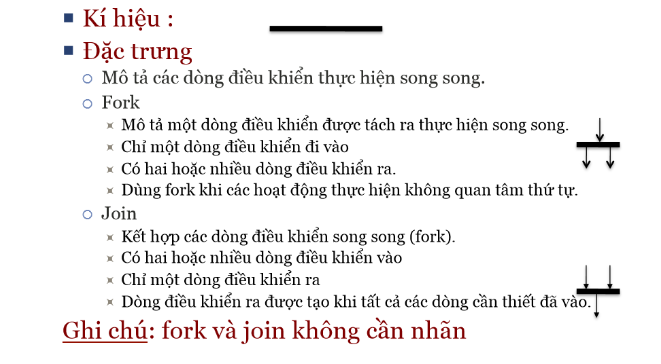


* + Transition:



* + Decisition:



* + Synchoronization bar: 
  + End:



* 1. Sơ đồ trình tự

Biểu đồ tuần tự là biểu đồ dùng để xác định các trình tự diễn ra sự kiện của một nhóm đối tượng nào đó. Nó miêu tả chi tiết các thông điệp được gửi và nhận giữa các đối tượng đồng thời cũng chú trọng đến việc trình tự về mặt thời gian gửi và nhận các thông điệp đó.

#### Các thành phần của biểu đồ tuần tự

* Đối tượng (object or class): biểu diễn bằng các hình chữ nhật



Figure 10 Object

* Đường đời đối tượng (Lifelines): biểu diễn bằng các đường gạch rời thẳng đứng bên dưới các đối tượng



Figure 11 Đường đời đối tượng

* Thông điệp (Message): biểu diễn bằng các đường mũi tên



Figure 12 Message

Các loại thông điệp:

* Thông điệp đồng bộ (Synchronous Message): Thông điệp đồng bộ cần có một request trước hành động tiếp theo.



Figure 13 Synchoronous Message

* Thông điệp không đồng bộ (Asynchronous Message): Thông điệp không đồng bộ không cần có một request trước hành động tiếp theo.



Figure 14 Asynchronous Message

* Thông điệp chính mình (Self Message): Là thông điệp mà đối tượng gửi cho chính nó để thực hiện các hàm nội tại.



Figure 15 Self Message

* Thông điệp trả lời hoặc trả về (Reply or Return Message): Là thông điệp trả lời lại khi có request hoặc sau khi kiểm tra tính đúng đắn của một điều kiện nào đó.



Figure 16 Reply or Return Message

* Thông điệp tạo mới (Create Message): Là thông điệp được trả về khi tạo mới một đối tượng.



Figure 17 Create Message

* Thông điệp xóa (Delete Message) Là thông điệp được trả về khi xóa một đối tượng.



Figure 18 Delete Massage

Nguồn: <https://viblo.asia/p/phan-tich-thiet-ke-he-thong-thong-tin-su-dung-bieu-do-uml-phan-1-PjxMe6yNG4YL>

<https://viblo.asia/p/phan-tich-thiet-ke-he-thong-thong-tin-su-dung-bieu-do-uml-phan-2-0bDM6wpAG2X4>